UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/209,015	12/10/1998	NATHAN ABRAMSON	101.957.156	8933

7590 01/11/2008

MICHAEL A DIENER
HALE AND DORR
60 STATE STREET
BOSTON, MA 02109

EXAMINER	
RIES, LAURIE ANNE	
ART UNIT	PAPER NUMBER
2176	
MAIL DATE	DELIVERY MODE
01/11/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/209,015	ABRAMSON ET AL.	
	Examiner	Art Unit	
	Laurie Ries	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,10-12,14 and 15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4,10-12,14 and 15* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is a result of the discovery of newly found prior art that is pertinent to the instant claims, namely claims 1-4, 10-12, and 14-15 (See also Decision by the Board of Patent Office Appeals and Interferences, rendered 31 July 2007, and Amendment, filed by Applicant on 29 November 2007).

2. Pursuant to said Board decision, Applicant has cancelled claims 9 and 13. Applicant has rewritten claim 9 in independent form incorporating the limitations of claim 1. Applicant has also rewritten claim 13 in independent form incorporating the limitations of claim 12. Claims 1-4, 10-12, and 14-15 are pending. Claims 1 and 12 are independent claims.

3. It is noted that two independent sets of rejections are applied to the pending claims (the first set begins at paragraph 5, and the second set begins at paragraph 6).

4. It is also noted that a different Examiner has been assigned to this case. For convenience, the Examiner has maintained the formatting of the previous rejection, where appropriate.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-4, 10-12, and 14-15 are rejected under 35 U.S.C. 103(a) as being unpatentable over WO 98/44695 to Apple Computer, Inc, hereafter referred to as "Apple", in view of Applicant's Admitted Prior Art, hereafter referred to as "Applicant's Specification".

Claim dependency has been indicated in parentheses () as shorthand, and as a convenience for Applicant and as a reminder that the rejection of a dependent claim *implicitly* incorporates all elements of the rationale of the rejected base claim, *supra.*

As per independent claim 1, Apple teaches the claimed method for mapping input fields in a hypertext document including emitting program code for mapping software:

mapping (claimed mapping refers to 'action bindings 402 consist of a mapping between an event that Applet 201 triggers, for example...and an action on a server' – page 23, lines 13-15) input field names ("name" – Table one; "INPUTFIELD")) in a hypertext document to component properties when the

hypertext document is rendered (refer to 'keys' that represent the data or state managed by applets 701 – page 23, lines 7-9; 'key is bound to a specific object or variable 704 in the server' – page 23, lines 10-12; (an example of a name for the variable is *approx.* line 10 on the TABLE, page 15; also "The dictionary {or snapshot of the keys and their current values} is used upon invocation of an action" – page 26, lines 7-10), the software component being a server based component ("application logic in the server" – Abstract);

providing the rendered hypertext document to a user (**HTML** – page 15)

receiving from the user input field data in a named input field ('inputfield' – TABLE - page 15); and

using the mapping to determine an appropriate component property ('declarations file...provide declarations for the tag...initialize instance variables of an object and provide runtime information' – page 15, lines 34-35, See also TABLE 2) for the named input field and to call component methods for processing the input field data ('method 703 on the server is invoked' – page 23, lines 18-19), the mapping being done such that the software component can process the input field regardless of the spelling of the name in the hypertext document (Apple does not put a restriction on the "name" field, this is processed regardless of its spelling).

Apple does not teach expressly converting the submitted input field data to a correct data type, however, Applicant's Specification notes that "a common technique for processing such input data is to create a software component that is specialized for each individual hypertext document, such as a component

designed to receive a book order. The software component parses the submitted data to search for the input fields with appropriate names...If the correctly named input field is found, the component must further convert the submitted data in the input field to the correct data type (See Applicant's Specification, "Background of the Invention", Page 2, lines 7-13, and Figure 1, noted as "Prior Art").

At the time of the invention it would have been obvious to one of ordinary skill in the art to apply the common technique of converting input field data to a correct type, as taught by Applicant's Specification, to the method for mapping input fields, as taught by Apple. The motivation for doing so would have been to ensure that the data is processed such that it can be properly displayed in a document; for instance, a "ZIP_CODE" field must be converted from text to integer and must contain five or nine numbers to be a valid United States zip code (See Applicant's Specification, Page 2, lines 15-18).

Therefore, it would have been obvious to combine Applicant's Specification with Apple for the benefit of ensuring that the data is processed such that it can be properly displayed in a document to obtain the invention as specified in claim 1.

As per dependent claim 2, Apple further teaches: wherein the rendering includes emitting hypertext form tags with a current value of an inputfield pre-filled in ('List applet' {a list of items}' – page 21, lines 10-11; 'pull-down lists' – page 6, line 7).

As per dependent claim 3, Apple further teaches : wherein the mapping includes encoding the hypertext input form with a unique name ('name' – Table

One) and registering the name (The WEBOBJECT tags in Table 1..."provide a pointer to entries n a declarations file" – page 15, lines 31-35; the name having a value of 'inputfield'...binds itself to the inputfield entry of the declarations file' – page 16, lines 32-35).

As per dependent claim 4, Apple lacks an explicit recitation of the receiving includes determining if the user submitted input field data is from a hypertext input form and bypassing input field processing if the determination cannot be made. However, it is suggested by Apple that "upon specified events on a browser by the user...certain actions may need to occur on a server". The present invention provides for the recognition of these events...and the invocation of the actions in the server. Therefore, since events are triggered by, for example, filling out a form, it is suggested that if the determination of the filling of the form cannot be completed, the action associated with the field will not be executed. It would therefore have been obvious to one of ordinary skill in the art at the time of the invention to bypass input field processing if not input to an HTML field is detected.

As per dependent claim 10, Apple further teaches wherein the determining includes iteratively processing input names associated with a component property to determine if data associated with any of the input names has been entered. (*inherent* in "Before an action is invoked in the server, any state that has changed in the client is transmitted back to the server {e.g. state may change when a browser user enters information in an input field of the web page} – page 20, lines 2—25. The "inheritance" relied upon is due to the

observation that CPU processed data in clock cycles, and the ONLY way for the CPU to detect a state change would have been to continually check for the state change; See also "a timer may be utilized such that synchronization is to occur every {5} seconds" – page 21, lines 1-3). Should Applicant dispute the inherence of the claimed iterations, it would have been obvious to one of ordinary skill in the art at the time of the invention to iterate Apple because it was suggested on Page 21, lines 1-3, and in order to ensure synchronization as described by Apple ("synchronization" - page 20, lines 19-23 *et seq.*)

As per dependent claim 11, Apple demonstrates all elements as applied in the rejection of independent claim 10, *supra*. "wherein the determining includes processing in order of priority stored with the mapping of the input field names", Apple lacks an explicit recitation of this feature. However, a priority of input fields is implied, because Action Coordinator 301 processed plural input fields, and because it processes "only those values that have changed since the last communication with the browser are compiled into the package" – page 29, lines 1-10. It is further implied in the observation that input field value mappings would not have been processed randomly, thus there is an inherent order. Since "priority" is generically claimed, it is believed that Apple implicitly meets this limitation. Otherwise, it would have been obvious to one of ordinary skill in the art at the time of the invention to process input fields in a prioritized order to give preference to the executions of certain actions over others.

As per independent claim 12, Apple teaches the claimed system for mapping hypertext input fields to software components comprising:

A preprocessor (In the reference as applied the claimed preprocessor is the CPU used to generate the code in Table One, e.g. WEBOBJECT name=INPUTFIELD></WEBOBJECT>...THE ABOVE HTML template includes tags for HTML, HEAD, BODY, and WEBOBJECT elements" – page 15, lines 23-25) for generating program code to register mappings between hypertext input field names and component properties and to emit hypertext form tags (authoring tool inherent in Apple's teaching relied upon, *supra*);

A name-space manager (*i.e.* Associations 302 – page 22, line 9 *et seq.* in conjunction with Action Coordinator 301; See also Applet group Controller – page 8, lines 19-20; "declarations file" - page 17, line 15 *et seq.*) for registering the mappings (Associations 302 and applets 201 include the ability to obtain the applet's keys, obtain the key values – page 22, lines 19-20; State bindings 401 include a list of 'keys' that represent the data or state managed by applets 701..."a key is bound to a specific object or variable 704 in the server to which it is synchronized" – page 23, lines 10-11; "Initially, when page is generated, all the state [sic.] for which there are state bindings are sent to the client...This initial synchronization ensures that the server's data is used to initialize the web page" – page 20, lines 15-20; "initialize instance variables of an object" – page 15, line 35; and rendering the document so that the document can be provided to a user (Table One);

a data handler (Applet Group Controller, Action Controller 301), responsive to submitted input data with an input field name submitted by a user, for using the mappings to submitted input data with component properties

("HTML elements {including applets} are mapped to objects in an object-oriented environment" – page 14, lines 9-11), and for calling appropriate component methods for processing the input data ("The server invokes the appropriate functions using the values transmitted from the Action Coordinator" – page 9, lines 3-5) the mapping being done such that the software component can process the input field regardless of the spelling of the name in the hypertext document (Apple does not put a restriction on the "name" field, this is processed regardless of its spelling).

Apple does not teach expressly that the data handler converts the submitted input field data to a correct data type, however, Applicant's Specification notes that "a common technique for processing such input data is to create a software component that is specialized for each individual hypertext document, such as a component designed to receive a book order. The software component (i.e. data handler) parses the submitted data to search for the input fields with appropriate names...If the correctly named input field is found, the component (i.e. data handler) must further convert the submitted data in the input field to the correct data type (See Applicant's Specification, "Background of the Invention", Page 2, lines 7-13, and Figure 1, noted as "Prior Art").

At the time of the invention it would have been obvious to one of ordinary skill in the art to apply the common technique of converting input field data to a correct type, as taught by Applicant's Specification, to the method for mapping input fields, as taught by Apple. The motivation for doing so would have been to ensure that the data is processed such that it can be properly displayed in a

document; for instance, a "ZIP_CODE" field must be converted from text to integer and must contain five or nine numbers to be a valid United States zip code (See Applicant's Specification, Page 2, lines 15-18).

Therefore, it would have been obvious to combine Applicant's Specification with Apple for the benefit of ensuring that the data is processed such that it can be properly displayed in a document to obtain the invention as specified in claim 12.

As per dependent claim 14, Apple further suggests claimed: "wherein the name-space manager includes a table for mapping a form to input field" (claimed table is *implicit* in mapping), and for mapping input fields to a component property (*idem*). Should Applicant disagree that Apple explicitly teaches this claim limitation; "Official Notice" is hereby taken that is was notoriously well-known to map using a table. It would therefore have been obvious to one of ordinary skill in the art at the time of the invention to employ tables in the mapping of Apple in order to efficiently perform the look-up function inherent in the mapping of Apple, in order to reduce space.

As per dependent claim 15, Apple further suggests wherein the name-space manager includes a *table* for mapping a form to input fields as described in detail in the rejection of dependent claim 14, *supra*. and for mapping input fields to a priority that determines the order in which the data handler processes the input field mappings. However, per prioritizing the processing of mapping of the input field names", Apple lacks an explicit recitation of this feature. A priority of input fields is implied, because Action Coordinator 301 processes plural input

fields, and because it processes "only those values that have changed since the last communication with the browser are compiled into the package" - page 29, lines 1-10. It is further implied in the observation that input field value mappings would not have been processed randomly, thus there is an inherent order. Since "priority" is generically claimed, it is believed that Apple implicitly meets this limitation. Otherwise, it would have been obvious to one of ordinary skill in the art at the time of the invention to process input fields in a prioritized order in order to give preference to the executions of certain actions over others.

6. Claims 1-4, 10-12, and 14-15 are rejected under 35 U.S.C. 103(a) as being unpatentable over WO 98/44695 to Apple Computer, Inc, hereafter referred to as "Apple", in view of Diedrich (U.S. Patent 6,173,288 B1).

As per independent claim 1, Apple teaches the claimed method for mapping input fields in a hypertext document including emitting program code for mapping software:

mapping (claimed mapping refers to 'action bindings 402 consist of a mapping between an event that Applet 201 triggers, for example...and an action on a server' – page 23, lines 13-15) input field names ("name" – Table one; "INPUTFIELD")) in a hypertext document to component properties when the hypertext document is rendered (refer to 'keys' that represent the data or state

managed by applets 701 – page 23, lines 7-9; 'key is bound to a specific object or variable 704 in the server' – page 23, lines 10-12; (an example of a name for the variable is *approx.* line 10 on the TABLE, page 15; also "The dictionary {or snapshot of the keys and their current values} is used upon invocation of an action" – page 26, lines 7-10), the software component being a server based component ("application logic in the server" – Abstract);

providing the rendered hypertext document to a user (**HTML** – page 15)

receiving from the user input field data in a named input field ('inputfield' – TABLE - page 15); and

using the mapping to determine an appropriate component property ('declarations file...provide declarations for the tag...initialize instance variables of an object and provide runtime information' – page 15, lines 34-35, See also TABLE 2) for the named input field and to call component methods for processing the input field data ('method 703 on the server is invoked' – page 23, lines 18-19), the mapping being done such that the software component can process the input field regardless of the spelling of the name in the hypertext document (Apple does not put a restriction on the "name" field, this is processed regardless of its spelling).

Apple does not teach expressly converting the submitted input field data to a correct data type, however, Diedrich teaches converting input data to a correct data type and size (See Diedrich, Figure 4, element 212, and Colum 3, lines 41-43).

Apple and Diedrich are analogous art because they are from the same field of endeavor of processing input data.

At the time of the invention it would have been obvious to one of ordinary skill in the art to apply the converting of input field data to a correct type, as taught by Diedrich, to the method for mapping input fields, as taught by Apple. The motivation for doing so would have been to ensure that the data is processed such that it can be properly displayed in a document; for instance, a "ZIP_CODE" field must be converted from text to integer (i.e. data type) and must contain five or nine numbers to be a valid United States zip code (i.e. data size).

Therefore, it would have been obvious to combine Diedrich with Apple for the benefit of ensuring that the data is processed such that it can be properly displayed in a document to obtain the invention as specified in claim 1.

As per dependent claim 2, Apple further teaches: wherein the rendering includes emitting hypertext form tags with a current value of an inputfield pre-filled in ('List applet' {a list of items}' – page 21, lines 10-11; 'pull-down lists' – page 6, line 7).

As per dependent claim 3, Apple further teaches : wherein the mapping includes encoding the hypertext input form with a unique name ('name' – Table One) and registering the name (The WEBOBJECT tags in Table 1..."provide a pointer to entries n a declarations file" – page 15, lines 31-35; the name having a value of 'inputfield'...binds itself to the inputfield entry of the declarations file' – page 16, lines 32-35).

As per dependent claim 4, Apple lacks an explicit recitation of the receiving includes determining if the user submitted input field data is from a hypertext input form and bypassing input field processing if the determination cannot be made. However, it is suggested by Apple that "upon specified events on a browser by the user...certain actions may need to occur on a server". The present invention provides for the recognition of these events...and the invocation of the actions in the server. Therefore, since events are triggered by, for example, filling out a form, it is suggested that if the determination of the filling of the form cannot be completed, the action associated with the field will not be executed. It would therefore have been obvious to one of ordinary skill in the art at the time of the invention to bypass input field processing if not input to an HTML field is detected.

As per dependent claim 10, Apple further teaches wherein the determining includes iteratively processing input names associated with a component property to determine if data associated with any of the input names has been entered. (*inherent* in "Before an action is invoked in the server, any state that has changed in the client is transmitted back to the server {e.g. state may change when a browser user enters information in an input field of the web page} – page 20, lines 2—25. The "inheritance" relied upon is due to the observation that CPU processed data in clock cycles, and the ONLY way for the CPU to detect a state change would have been to continually check for the state change; See also "a timer may be utilized such that synchronization is to occur every {5} seconds" – page 21, lines 1-3). Should Applicant dispute the inherence

of the claimed iterations, it would have been obvious to one of ordinary skill in the art at the time of the invention to iterate Apple because it was suggested on Page 21, lines 1-3, and in order to ensure synchronization as described by Apple ("synchronization" - page 20, lines 19-23 *et seq.*)

As per dependent claim 11, Apple demonstrates all elements as applied in the rejection of independent claim 10, *supra*. "wherein the determining includes processing in order of priority stored with the mapping of the input field names", Apple lacks an explicit recitation of this feature. However, a priority of input fields is implied, because Action Coordinator 301 processed plural input fields, and because it processes "only those values that have changed since the last communication with the browser are compiled into the package" – page 29, lines 1-10. It is further implied in the observation that input field value mappings would not have been processed randomly, thus there is an inherent order. Since "priority" is generically claimed, it is believed that Apple implicitly meets this limitation. Otherwise, it would have been obvious to one of ordinary skill in the art at the time of the invention to process input fields in a prioritized order to give preference to the executions of certain actions over others.

As per independent claim 12, Apple teaches the claimed system for mapping hypertext input fields to software components comprising:

A preprocessor (In the reference as applied the claimed preprocessor is the CPU used to generate the code in Table One, e.g. WEBOBJECT name=INPUTFIELD></WEBOBJECT>...THE ABOVE HTML template includes tags for HTML, HEAD, BODY, and WEBOBJECT elements" – page 15, lines 23-

25) for generating program code to register mappings between hypertext input field names and component properties and to emit hypertext form tags (authoring tool inherent in Apple's teaching relied upon, *supra*);

A name-space manager (*i.e.* Associations 302 – page 22, line 9 *et seq.* in conjunction with Action Coordinator 301; See also Applet group Controller – page 8, lines 19-20; "declarations file" - page 17, line 15 *et seq.*) for registering the mappings (Associations 302 and applets 201 include the ability to obtain the applet's keys, obtain the key values – page 22, lines 19-20; State bindings 401 include a list of 'keys' that represent the data or state managed by applets 701..."a key is bound to a specific object or variable 704 in the server to which it is synchronized" – page 23, lines 10-11; "Initially, when page is generated, all the state [sic.] for which there are state bindings are sent to the client...This initial synchronization ensures that the server's data is used to initialize the web page" – page 20, lines 15-20; "initialize instance variables of an object" – page 15, line 35; and rendering the document so that the document can be provided to a user (Table One);

a data handler (Applet Group Controller, Action Controller 301), responsive to submitted input data with an input field name submitted by a user, for using the mappings to submitted input data with component properties ("HTML elements {including applets} are mapped to objects in an object-oriented environment" – page 14, lines 9-11), and for calling appropriate component methods for processing the input data ("The server invokes the appropriate functions using the values transmitted from the Action Coordinator" – page 9,

lines 3-5) the mapping being done such that the software component can process the input field regardless of the spelling of the name in the hypertext document (Apple does not put a restriction on the "name" field, this is processed regardless of its spelling).

Apple does not teach expressly converting the submitted input field data to a correct data type, however, Diedrich teaches that the CGI programs of the preferred embodiment of the invention convert input data to a correct data type and size (See Diedrich, Figure 4, element 212, Column 2, lines 23-32, and Colum 3, lines 41-43).

Apple and Diedrich are analogous art because they are from the same field of endeavor of processing input data.

At the time of the invention it would have been obvious to one of ordinary skill in the art to apply the converting of input field data to a correct type, as taught by Diedrich, to the method for mapping input fields, as taught by Apple. The motivation for doing so would have been to ensure that the data is processed such that it can be properly displayed in a document; for instance, a "ZIP_CODE" field must be converted from text to integer (i.e. data type) and must contain five or nine numbers to be a valid United States zip code (i.e. data size).

Therefore, it would have been obvious to combine Diedrich with Apple for the benefit of ensuring that the data is processed such that it can be properly displayed in a document to obtain the invention as specified in claim 12.

As per dependent claim 14, Apple further suggests claimed: "wherein the name-space manager includes a table for mapping a form to input field" (claimed table is *implicit* in mapping), and for mapping input fields to a component property (*idem*). Should Applicant disagree that Apple explicitly teaches this claim limitation; "Official Notice" is hereby taken that is was notoriously well-known to map using a table. It would therefore have been obvious to one of ordinary skill in the art at the time of the invention to employ tables in the mapping of Apple in order to efficiently perform the look-up function inherent in the mapping of Apple, in order to reduce space.

As per dependent claim 15, Apple further suggests wherein the name-space manager includes a *table* for mapping a form to input fields as described in detail in the rejection of dependent claim 14, *supra*. and for mapping input fields to a priority that determines the order in which the data handler processes the input field mappings. However, per prioritizing the processing of mapping of the input field names", Apple lacks an explicit recitation of this feature. A priority of input fields is implied, because Action Coordinator 301 processes plural input fields, and because it processes "only those values that have changed since the last communication with the browser are compiled into the package" - page 29, lines 1-10. It is further implied in the observation that input field value mappings would not have been processed randomly, thus there is an inherent order. Since "priority" is generically claimed, it is believed that Apple implicitly meets this limitation. Otherwise, it would have been obvious to one of ordinary skill in the

art at the time of the invention to process input fields in a prioritized order in order to give preference to the executions of certain actions over others.

Response to Arguments

7. Applicant's arguments with respect to claims 1-4, 10-12, and 14-15 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Laurie Ries whose telephone number is (571) 272-4095. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Doug Hutton, can be reached at (571) 272-4137.

9 Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through

Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

LR

/William L. Bashore/
Primary Examiner
Tech Center 2100

WENDY GARBER DIRECTOR
TECHNOLOGY CENTER 2100